On the Effect of Head Tagging on Parsing Discontinuous Dependencies in French

Ophélie Lacroix

On the Effect of Head Tagging on Parsing Discontinuous Dependencies in French

Ophélie Lacroix

LINA - Université de Nantes, 2 Rue de la Houssinière, 44322 Nantes Cedex 3
ophelie.lacroix@univ-nantes.fr

Abstract. In this paper we aim at showing the strong impact of head tagging on syntactic dependency parsing. The rules of categorial dependency grammar used to parse French deal with discontinuous dependencies and long distance syntactic relations. Such parsing method produces a substantial number of dependency structures and requires too much parsing time. We show that a local tagging method can reduce these problems and help to solve the global problem of dependency parsing disambiguation. Then we adapt a tagging method (CRF) to types of the categorial dependency grammar. We obtain a dependency head preselection allowing us to reduce parsing ambiguity and to see that we can find distant relation of dependencies through local results of such method.

1 Introduction

Syntactic parsing is a well-known task in natural language processing. Its purpose is to attach syntactic structure to a sentence using a grammar. There are several types of syntactic structures. Among the syntactic structures used the most popular are the constituency and dependency structures. We choose in this work the dependency structures [13,20] because the order in French is flexible and admits many discontinuous constructions which are represented by discontinuous dependencies. The dependency representation allows us to highlight the syntactic functions binding words of the sentences. Notably, figure 1 presents the non-projective comparative relation *moins que* (less than). A dependency d is a binary relation between a governor g and a subordinate s ($g \xrightarrow{d} s$). We call head dependency type (head type) the name of the dependency pointing on the subordinate. Dependencies can be projective or discontinuous [1]. Discontinuous dependencies have always been an obstacle for dependency parsing. This is why we choose the class of categorial dependency grammar (CDG) [4,8] which resolve this problem. We use in this work the CDG of French [11] developed using CDG Lab [1]. The CDG Lab is an integrated environment for elaboration and maintainance of CDG grammars and corpora of dependency trees. In the CDG

[1] A dependency $g \to s$ is discontinuous if a word between the governor and the subordinate does not depend on the governor. These dependencies can cross others.

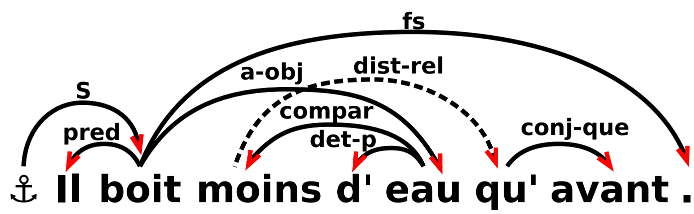

Fig. 1. *Dependency structure for the French sentence* "Il boit moins d'eau qu'avant." *(He drinks less water than before).*

Lab is integrated a parser for CDG. It allows 3 different methods of parsing : *autonomous parsing*, *parsing by head type selection* and *parsing by approximation*. Parsing by head type selection consists in selecting the proper head dependency type for every word and computing the structure conforming to these choices. In the head type selection mode, the head types are selected manually. This method drastically reduces the ambiguity of parsing. Our objective is to replace this manual choice by a stochastic procedure. So this objective may be seen as a problem of tagging the text with head types. First of all, this tagging should be adapted to the categorial dependency grammar of French we use. This task can be seen as a kind of supertagging [2] which assigns complex tags (with a rich description) on lexical units in order to disambiguate parsing. It is known to reduce ambiguity and parsing time. Here the difficulty of tagging consists of locally computing the correct head type for distant dependencies. The main difficulty of such tagging is to properly compute the head type not only in the case when the governor is local but even when it is distant. Very many methods have been developed for pos-tagging. Some of them improving dependency parsing [14,5]. In this parper, we choose CRF techniques and tools [2] to train a French dependency tag model using the corpora developed in CDG Lab. Then, the tagging results are used in the place of manual head selection for parsing French sentences. In this paper, we statistically show the substantial effect of head type selection on parsing time and on the number of sucessful parses. We compare these results with results of autonomous parsing.
This work is preliminary because the head type selection procedure uses a good splitting of the text into composite lexical units and also a tagging of morphosyntactic classes. So this is a part of a larger project where all these preliminary information will be computed by other stochastic procedures.

2 Categorial Dependency Grammar

The categorial dependency grammars [7,10] are rather close to the classical categorial grammars [3]. The novelty are the polarized valencies used to define discontinuous dependencies. The projective dependencies defined using the classical

[2] Conditional Random Fields.

categories are constructed through the classical type elimination rules whereas the discontinuous dependencies are created through pairing of dual polarities ($\nearrow\searrow$ and $\swarrow\nwarrow$). The rules of the CDG are presented in table 1.

$\mathbf{L}^1$ $C^{P_1}\,[C\backslash\beta]^{P_2} \vdash [\beta]^{P_1P_2}$
$\mathbf{I}^1$ $C^{P_1}\,[C^*\backslash\beta]^{P_2} \vdash [C^*\backslash\beta]^{P_1P_2}$
Ω^1 $[C^*\backslash\beta]^{P} \vdash [\beta]^{P}$
$\mathbf{D}^1$ $\alpha^{P_1(\swarrow C)P(\nwarrow C)P_2} \vdash \alpha^{P_1PP_2}$, if the potential $(\swarrow C)(\nwarrow C)$ satisfies the pairing rule **FA**.

Table 1. *Left rules of CDG. Right rules are symetrical. The **L**, **I** et Ω are used to eliminate classical categories and iterable categories (i$\geq$ 0 times derivable) [9]. They create the projective dependencies. In the same time, the polarized valencies (P_1, P_2) are concatenated in a string called potential. The **D** rule allows to eliminate valencies according to the **FA** principle : the nearest valencies are eliminate first. This rule creates the discontinuous dependency C.*

2.1 Grammar and Dependency Corpus for French

To parse French we use the categorial dependency grammar of French [11]. It uses 117 French dependency types (corresponding to head types) distributed over 39 dependency groups. Then, we called head group the dependency group which can be derived from head type. As a matter of example, the dependency *a-obj* is the head type for the direct object (the object in accusative case), *d-obj* and *g-obj* are head types of indirect objects (respectively in dative and genitive cases). All the 3 are elements of the group OBJ. On the other hand, among the dependency types there are 27 discontinuous dependencies such as *clitic*, *modifier*, *reflexive*, *apposition*, *negative* and some other discontinuous dependencies.

We use a corpus counting 2778 dependency structures [3] over 35203 lexical units. The corpus was developed and is exploited by the members of the team TALN [4] of the LINA laboratory. We notice, though only 4% of dependencies in the corpus are discontinuous but 41% of dependency structures have at least one discontinuous dependency. So discontinuous dependencies are well enough represented in this French corpus. In this corpus, not only every lexical unit has a head type (belong into a head group) but it is also categorized by one of the 185 morphosyntactic classes of the categorial dependency grammar of French. We use 2 levels of the categorization: a more general one using 28 tags (hyper classes) and a more specific one using 86 tags (extended classes). The hyper morphosyntactic tagging gives a rudimentary lexical syntactic information whereas

[3] 2778 French sentences.
[4] The corpus is not publicly available yet.

the extended one adds some semantic syntactically relevant information. As an example in figure 1, *moins* functions as an adverb in the lexical unit *moins que* (less than). It is tagged by **Adv** using the hyper tagging whereas it may be tagged with **Adv-compar** using the extended tagging. The lexical unit *que* is a junction tagged with **Conj** using the hyper tagging and which may be **Conj-compar** using the extended. This extended tagging shows that both members of the dependency are related through comparison. Figure 2 illustrate the information recovered from the annotated dependency structures for a particular sentence.

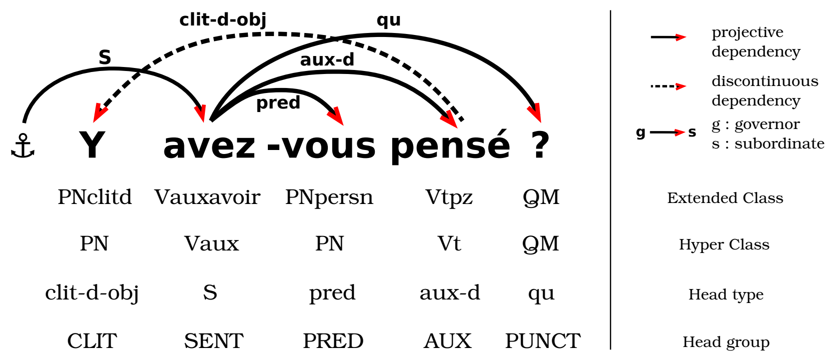

Fig. 2. *Dependency structure for the French sentence* "Y avez vous pensé?" *(Have you thought about it?) and hyper classes, extended classes, head types and head groups of each lexical unit.*

2.2 CDG Lab

As we have already mentioned, the CDG Lab parser has 3 complementary parsing methods :

– *autonomous parsing* mode applied to raw french sentences. It computes both the correct head type and the proper morphosyntactic class. The sentence is split into lexical units and to this splitting is applied a CKY-based algorithm which searches all possible correct dependency structures. This parser treats a huge number of variants ;
– *parsing by head type selection* is a semi-automatic mode. The user selects grouping lexical units into composite lexical units, selects proper morphosyntactic classes for them and head types (or head groups). Then the parser compute the dependency structures compatible with this selection. This mode drastically reduces the number of choices in the analysis ;
– *parsing by approximation* mode is performed after the preliminary head type (or head group) selection. For every output dependency structure the user

can positively or negatively annotate the dependencies and the parser re-analyses the sentence using these annotations.

Here, we consider the head type (or head group) selection mode. We observe that the manual tagging reduces the ambiguity of analyses. Generally, it garantees a very restricted number of solutions compatibles with the selection. The other benefit of this mode is a significant reduction of parsing time. By replacing the manual head type selection by an automatic head type selection we hope to attain both advantages. In this work, we focus on head type (and head group) tagging, which means that we know the lexical units decomposition and their morphosyntactic categorization on the training and tagging phase.

3 Head type Tagging

The most popular tagging procedures are pos-tagging and morphosyntactic tagging, both different from the head type tagging. Nevertheless, the tools for all of them may be identical. Among the various methods of tagging, the most popular are the stochastic graphical models such as Hidden Markov Model (HMM) [18], Maximum Entropy Model (MEMM) [19] and Conditional Random Fields (CRF) [17]. In this work, we choose the CRF method because it is able to deal with a large number of tags and features and consider the notion of sequence.

3.1 Software and Feature Patterns

We use the Wapiti software [12] to train our dependency model for tagging sentences. We use 15 features which apply to lexical units and their morphosyntactic classes. The window size we consider is of 5 lexical units containing the current one. For the classes we also have a window of 7 (both for hyper and extended classes). Features also consider the word suffixes and capital letters.

3.2 Experimentation and Evaluation

To evaluate the tagging we divide the corpus in 10 parts. Each experiment includes a training phase performed on 90% of sentences of the corpus and the tagging phase performed on the last 10%. Wapiti can yield the n best tagging sequences for each sentence: we choose to produce the 10 best. Then, for each lexical unit we evaluate the taggings with 1, 2, 5 or 10 best tags assigned. Sequences can be very similar from one to another, there are rarely 10 different tags for a lexical unit. We should decide whether among the 1, 1 to 2, 1 to 5 or 1 to 10 assigned tags there is the right one, that is we evaluate the accuracy on the top 1, 2, 5 and 10. The results of the evaluation are presented in table 2.

One may observe that when the number of assigned tags (per lexical unit) increases, the accuracy increases. There are more possibilities to have the right tag among various possibilities. And the head group tagging gives best results

using	Head type tagging		Head group tagging	
	hyper classes	extended classes	hyper classes	extended classes
Top 1	87.8	91.1	90.4	91.6
Top 2	90.0	93.2	92.5	93.7
Top 5	92.9	95.5	95.1	96.0
Top 10	94.6	96.6	96.4	97.1

Table 2. *Evaluation (accuracy) of head type and group tagging using Wapiti. Results represent the rate of lexical units correctly tagged among the 1, 1 to 2, 1 to 5 or 1 to 10 assigned tags.*

because the ambiguity of tagging is minor. As it is shown in table 3, the average number of possible head groups is lower than that of head types for a given class. The head groups score is not significantly better but can make a difference for parsing because assigning a head group to a lexical unit is equivalent to assigning the disjunction of the head types in this group. For example, when we assign to a lexical unit the group OBJ we actually assign the disjunction of 7 head types [5]. We will observe this difference through parsing evaluation. Moreover, the extended classes bring information improving the tagging of head types and groups. Then we choose to perform parsing using head types and head groups found using extended classes.

Average number of	head types	(max.)	head groups	(max.)
Per hyper class	13	(43)	7	(18)
extended class	6	(31)	4	(16)

Table 3. *Average (and maximum) number of possible head types and head groups for a given hyper or extended class.*

4 Dependency parsing and Evaluation

4.1 Parsing and evaluation process

Our objective is to replace the manual head type selection mode by the automatic selection using the CRF trained model. As a result of tagging, we have a corpus where each lexical unit is tagged by 1 to at most 10 head types or head groups. We parse all of the corpus using these head type or head group pre-selections. Both pre-selections reduce the ambiguity and so the number of

[5] *a-obj, o-obj, d-obj, l-obj, g-obj, a-obj-d, qa-obj.*

the output dependency structures for each sentence. One of particular goals is to know whether among the dependency structures there is the best one [6]. Currently, the output dependency structures are not sorted by the distance from the best one. So we use the best dependency structure to find the nearest output dependency structure. Nearest means having the maximum of correctly labelled dependencies as compared to the best one. We show the phases of this entire procedure in figure 3.

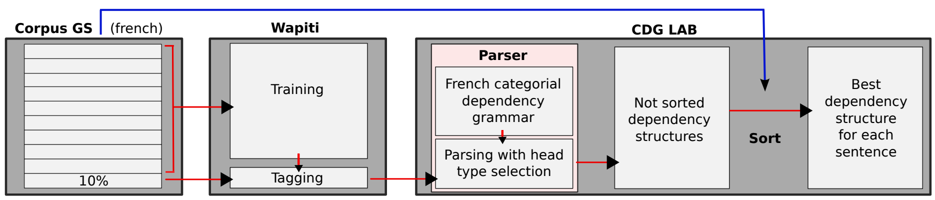

Fig. 3. *Proceeding diagram. Training is performed on 90% of the corpus and tagging on the remaining 10%. The tagged part is parsed using the head type selection. The parser output several unsorted dependency structures and finds the nearest to the best dependency structure for each sentence.*

For each sentence, the results are evaluated only considering the nearest dependency structure. By the way not all sentences are successfully analysed and obtain a dependency structure. There are 2 reasons for that :

– the assigned head types (or head groups) contradict to the grammar categories, so the parsing fails ;
– the parsing is avoided because of time limits (often because of the length of the sentence).

As a consequence, one of the evaluation criteria is the number of completed parses.

4.2 Results and discussion

First of all, we find the rate of sentences obtaining at least one dependency structure are the completed parses. And we are interested to see the evolution of parsing time and the average length of sentences for the completed and uncompleted parses. The experimentation results are presented in Table 4. On that one hand, these results allow to compare the performance of the autonomous parsing and of the parsing by head type selection. The important result is the impressive reduction of the parsing time. As a consequence, the rate of sentences for which the parsing fails because of time limit is close to zero. These few uncompleted parses correspond to very long sentences (> 41 lexical units per sentence). Note

[6] The best dependency structure is that one associated to the sentence in the corpus.

Autonomous parsing

Nb of types	Number of sentences						LU/sentence			Parsing time	
	CP	(%)	UP-C	(%)	UP-T	(%)	CP	UP-C	UP-T	CP	UP
-	1150	(41.4)	3	(00.1)	1625	(58.5)	7.2	7.3	17.6	42min24	4h30

Parsing by head type selection

Nb of types	Number of sentences						LU/sentence			Parsing time	
	CP	(%)	UP-C	(%)	UP-T	(%)	CP	UP-C	UP-T	CP	UP
1	1805	(65.0)	969	(34.9)	**4**	(00.1)	11.5	16.5	52.5	3min03	1min35
1 to 2	2054	(73.9)	718	(25.8)	6	(00.2)	11.6	17.7	56.1	4min16	1min53
1 to 5	2335	(84.1)	438	(15.8)	5	(00.2)	12.0	20.0	49.4	6min02	1min29
1 to 10	2505	(90.2)	262	(09.4)	11	(00.4)	12.2	22.5	42.8	8min01	2min23

Parsing by head group selection

Nb of groups	Number of sentences						LU/sentence			Parsing time	
	CP	(%)	UP-C	(%)	UP-T	(%)	CP	UP-C	UP-T	CP	UP
1	1931	(69.5)	832	(29.9)	15	(00.5)	11.5	16.9	45.8	6min41	3min31
1 to 2	2172	(78.2)	586	(21.1)	20	(00.7)	11.6	18.6	45.0	8min52	4min15
1 to 5	2439	(87.8)	302	(10.9)	37	(01.3)	11.8	21.6	43.6	12min05	6min47
1 to 10	**2548**	(91.7)	**179**	(06.4)	51	(01.8)	12.0	24.4	41.6	16min43	9min03

Table 4. *Number of completed parses (CP), number of uncompleted parses because head selection was not consistent with grammar's rules (UP-C) or because of parsing time (UP-T). Parsing time is limited to 10s. Average number of lexical units (LU) per sentence in the three cases. Total parsing time for completed parses and both uncompleted parses.*

that there are fewer parses uncompleted because of time limits, but more parses uncompleted because of grammatical conflicts. Indeed, the wrong selection of head types or head groups can make fail the parsing because the sequence is ungrammatical.

From the results on parsing by head type and head group selection, one may observe that the number of completed parses increases when increases the number of possible head types or head groups. There are more chances to obtain the right head type or head group among various tagging possibilities. So the number of conflicts with the grammar categories decreases and the number of completed parses increases. The best results (for 10 possible head groups) gives 2548 completed parses among 2778 (91.7%), among which 2088 give the best dependency structure. Moreover, we saw that the head type and head group selections highly reduce the parsing time. At the same time it may increase with the growing number of alternative tags. The reason is clear : the more alternative head types or head groups are selected the more dependency structures are produced for one sentence. So for very long sentences the parsing fails with few head type or head group selection, but succeeds with more. Respectively, the head group selection allows to parse more sentences but takes more time, because the groups often include several head types. So using the groups, we obtain more parsed sentences even though some parses are stopped because of time limit.

Autonomous parsing

Nb of types	All dependencies		Discontinuous dependencies	
	LAS	UAS	LAS	UAS
-	98.3	99.0	92.7	93.2

Parsing by head type selection

Nb of types	All dependencies		Discontinuous dependencies	
	LAS	UAS	LAS	UAS
1	93.7	96.7	92.4	93.7
1 to 2	95.1	97.3	94.3	95.5
1 to 5	96.2	97.8	94.4	95.5
1 to 10	96.4	97.9	94.5	95.4

Parsing by head group selection

Nb of groups	All dependencies		Discontinuous dependencies	
	LAS	UAS	LAS	UAS
1	93.9	96.7	88.8	93.3
1 to 2	95.1	97.2	90.0	93.7
1 to 5	96.3	97.9	90.5	93.8
1 to 10	96.7	98.0	91.1	94.3

Table 5. *LAS and UAS for autonomous parsing compared to parsing by head selection. The evaluation is made on the best dependency structure (nearest to the original) produced by the parser for each completed parse. The labelled attachment score (LAS) is the rate of lexical units (apart from punctuations) bounded to the proper governor with the correct dependency label. The unlabelled attachment score (UAS) is the rate of lexical units bounded to the proper governor.*

The evaluation of the precision on the dependencies are presented in table 5. The evaluation is performed on completed parses and uses the labelled (LAS) and unlabelled (UAS) attachment score. Similarly, the results are better when there is a larger choice of head types or head groups. As above, the more there are head type or head group choices the more we have chances to have a right dependency. As it concerns the discontinuous dependencies, one can see that the precision is not as good as in the case of all dependencies (both for head types and head groups). Nevertheless, the scores rest interesting. In particular, the scores are better in the case of head type selection than in the case of head groups. The rate of discontinuous dependencies among all dependencies reach 4.3% to 4.6% with head type selection and 4.8% to 4.9% with head groups selection. We conjecture that some uncompleted parses of head types selection correspond to the difficult discontinuous cases. Sometimes, the more are possibilities of head type selection the easier is to find a parse. However, in some difficult discontinuous cases the right parses are not found. Nevertheless, parsing with head type (or head group) selection allows to find a good rate of correct dependencies (projective and discontinuous).

5 Related work

Here, we focus on one phase of dependency parsing. We based this work on a given morphosyntactic tagging and a given sorting of output dependency structures. That makes difficult the comparison with autonomous dependency parsing. Moreover, dependency parsing is often performed on the model derived from constituency parsing which does not reflect the non-projectivity. In a technically aspect, our work is close to supertagging which also reduces ambiguity and parsing time [6,15,16].

6 Conclusion and Future Work

We show that from different points of view, the head type tagging is very usefull for dependency parsing. The first point is that it really reduces the parsing time. A large number of sentences for which parsing failed in autonomous mode finally succeed with the head type selection. We also obtain a good rate of completed parses. Finally, the number of sentences obtaining at least one dependency structure is noticeably greater. This makes more interesting the good precision score because the high precision concerns more representative cases of parsing success. As a result, the head type tagging really ameliorates the dependency parsing. Nevertheless, this amelioration is attained using given correct splitting of the sentences into lexical units and given correct morphosyntactic classes. One may also add the given good sorting of dependency structures by their distance from the optimal one which makes the work easier. To arrive at a completely autonomous parsing, one is obliged to make these tasks automatically. The question is whether the head type tagging will be efficient enough? Another question is whether the recall in autonomous parsing will be high enough to parse a great part of French sentences. A possible idea to obtain better results would be to compute partial dependency structures after failure and to use them in evaluation.

References

1. Ramadan Alfared, Denis Béchet and Alexander Dikovsky. *"CDG Lab": a Toolbox for Dependency Grammars and Dependency Treebanks Development.* 272–281, Proceedings of DEPLING 2011, Barcelona, Spain (2011)
2. Srinivas Bangalore and Aravind K. Joshi. *Supertagging: Using Complex Lexical Descriptions in Natural Language Processing.* Mit Press (2010)
3. Y. Bar-Hillel, C. Gaifman and E. Shamir. *On Categorial and Phrase Structure Grammars.* 99–115, Language and information, Addison-Wesley (1964)
4. Denis Béchet, Alexander Dikovsky and Annie Foret. *Dependency Structure Grammar.* 18–34, Proceedings of LACL 2005, Bordeaux, France (2005)
5. Marie Candito, Benoît Crabbé and Pascal Denis. *Statistical French dependency parsing: treebank conversion and first results.* 1840–1847, Proceedings of LREC2010, La Valletta, Malta (2010)
6. Stephen Clark and James R. Curran. *The Importance of Supertagging for Wide-Coverage CCG Parsing.* 282–288, Proceedings of COLING 2004, Geneva, Switzerland (2004)
7. Michael Dekhtyar and Alexander Dikovsky. *Categorial Dependency Grammars.* 76–91, ICCG 2004, Montpellier, France (2004)
8. Michael Dekhtyar and Alexander Dikovsky. *Generalized Categorial Dependency Grammars.* 230-255, LNCS 4800, Springer (2008)
9. Michael Dekhtyar, Alexander Dikovsky and Boris Karlov. *Iterated Dependencies and Kleene Iteration.* 66–81, Formal Grammar 2010/2011, LNCS 7395 (2010)
10. Alexander Dikovsky. *Dependencies as Categories.* 90–97, Proceedings of COLING 2004 Workshop, Geneva, Switzerland (2004)
11. Alexander Dikovsky. *Categorial Dependency Grammars: from Theory to Large Scale Grammars.* 262–271, Proceedings of DEPLING 2011, Barcelona, Spain (2011)
12. Thomas Lavergne and Olivier Cappé and François Yvon. *Practical Very Large Scale CRFs.* Proceedings of ACL 2010, Uppsala, Sweden (2010)
13. Igor Mel'cuk. Dependency syntax : Theory and Practice. Mark Aronoff, State University of New York Press (1988)
14. Alexis Nasr. *Grammaires de dépendances génératives probabilistes. Modèle théorique et application à un corpus arboré du franais.* 115-153, Traitement Automatique des Langues volume 46 (2006)
15. Alexis Nasr and Owen Rambow. *Non-lexical chart parsing for TAG.* In [2] (2010)
16. Anoop Sarkar. *Combining Supertagging and Lexicalized Tree-Adjoining Grammar Parsing.* In [2] (2010)
17. Charles Sutton and Andrew McCallum. *An Introduction to Conditional Random Fields for Relational Learning.* Introduction to Statistical Relational Learning, MIT Press (2006)
18. Lawrence R. Rabiner. *A tutorial on hidden markov models and selected applications in speech recognition.* Proceedings of IEEE, (1989)
19. Adwait Ratnaparkhi. *A maximum entropy model for part-of-speech tagging.* Proceedings of EMNLP, University of Pennsylvania, USA (1996)
20. Lucien Tesnière. Éléments de syntaxe structurale. Klincksieck (1959)